

13. Конструкции за разклонение

Проф. д-р Емил Хаджиколев

1. Основни типове алгоритми и конструкции за реализацията им;
2. Условна конструкция if;
3. Условна конструкция if-else;
4. Вложени условни конструкции;
5. Конструкция за избор на вариант switch-case.

Основни типове алгоритми и езикови конструкции в езиците за програмиране (припомняне)

- **Линейни** – описаните в кода инструкции се изпълняват последователно;
- **Разклонени** – if, if-else, switch-case;
- **Циклични/итеративни** – for, do-while, while, for-each;
- **Рекурсивни** – в кода на подпрограма се извиква същата подпрограма (обикновено) с други параметри.

Условна конструкция if

- Чрез конструкцията **if** се указва **част от код**, който **се изпълнява само ако е изпълнено определено условие**.
- **Синтаксисът на конструкцията if** е следният:

```
if(<условие>){  
    [<тяло>]  
}
```
- **Условието** е логически израз.
- **Тялото** може да съдържа различни команди – изрази, конструкции и др. **То се изпълнява само ако стойността на условието е true**.
- Ако тялото се състои само от една команда, фигурните скоби може да се пропуснат. Това обаче не се препоръчва с цел по-добра четимост и сигурност при писане на кода.

Условна конструкция if - пример

```
#include <iostream>
#include <string>
using namespace std;

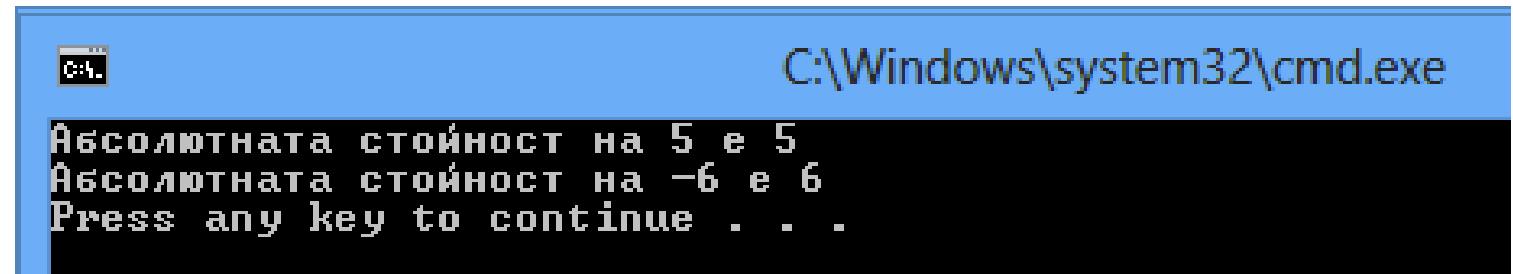
void getAbs(int i) {
    int absValue = i;
    if (i<0) {
        absValue = -i;
    }
    cout << "Абсолютната стойност на " << i << " е " << absValue << '\n';
}

int main(){
    setlocale(LC_ALL, "bg");

    getAbs(5);
    getAbs(-6);

    return 0;
}
```

В следващия пример е създадена функция, която получава като параметър число и извежда абсолютната му стойност.



Условна конструкция if-else

- При конструкция **if-else**, в зависимост от това, дали дадено условие е изпълнено, се изпълнява само единият от два блока с оператори.
- Синтаксисът на конструкцията if-else е:

```
if(<условие>){  
    [<тяло-if>]  
}else{  
    [<тяло-else>]  
}
```

- Ако условието има стойност **true** се изпълнява първият блок (тяло-if), иначе се изпълнява блокът, описан след **else** (тяло-else).

Условна конструкция if-else - пример

- Използването на if- else конструкция прави функцията getAbs по-лесна за четене:

```
void getAbs(int i) {  
    int absValue;  
  
    if (i<0) {  
        absValue = -i;  
    }  
    else {  
        absValue = i;  
    }  
  
    cout << "Абсолютната стойност на " << i << " е " << absValue << '\n';  
}
```

Пример за намиране на максималното от две числа

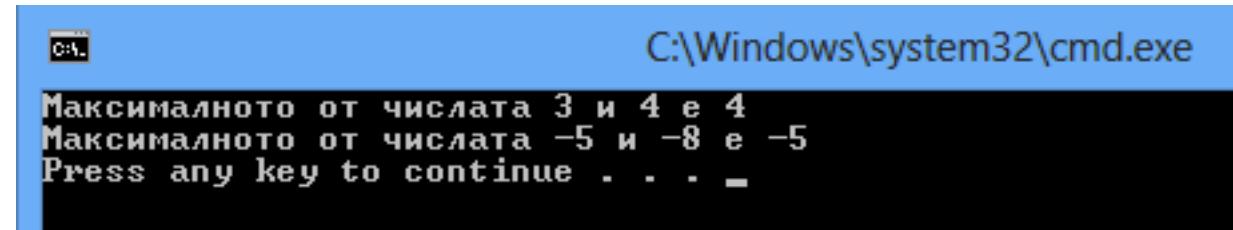
```
#include <iostream>
#include <string>
using namespace std;

void maxF(double d1, double d2) {
    double max;          // в max ще запомним максималното число
    if (d1>d2) {        // ако d1>d2 значи...
        max = d1;         // ... d1 е максималното,
    }
    else {               // иначе...
        max = d2;         // ... d2 е максималното
    }
    cout << "Максималното от числата " << d1 << " и " << d2 << " е " << max << '\n';
}

int main(){
    setlocale(LC_ALL, "bg");

    maxF(3, 4);
    maxF(-5, -8);

    return 0;
}
```



Вложени условни конструкции

- В множество задачи, **логиката на решението изискава използването на повече условия.**
- Например, ако за две цели числа $i1$ и $i2$ трябва да се изведе информация дали първото е по-малко, равно или по-голямо от второто, трябва да се направят поне две сравнения:
 - ако първото сравнение $i1 < i2$ е истина, ще се изведе съобщението „ $i1$ е по-малко от $i2$ ”,
 - иначе за да се определи кой от останалите два варианта е истина, трябва да извърши още едно сравнение – напр., $i1 == i2$, което е необходимо да се вложи в `else`-блока на първото.
 - Такова решение е показано в следващия пример.

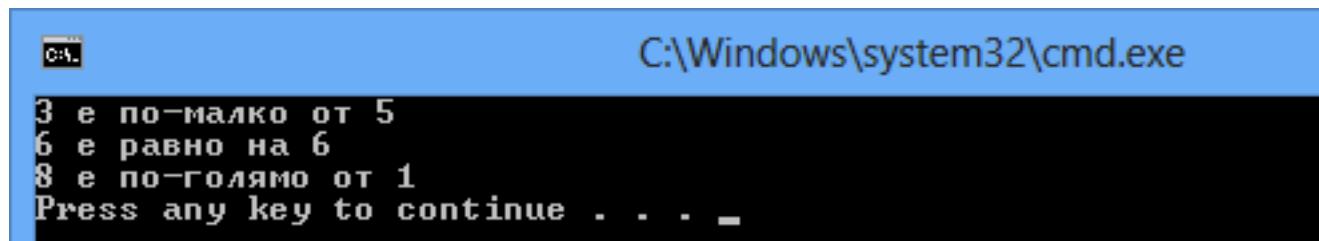
Пример

```
void comparisonInfo(int i1, int i2) {
    string info="";
    if (i1<i2) {
        info = to_string(i1) + " е по-малко от " + to_string(i2);
    }
    else {
        if (i1 == i2) {
            info = to_string(i1) + " е равно на " + to_string(i2);
        }
        else {
            info = to_string(i1) + " е по-голямо от " + to_string(i2);
        }
    }
    cout << info << '\n';
}

int main(){
    setlocale(LC_ALL, "bg");

    comparisonInfo(3, 5);
    comparisonInfo(6, 6);
    comparisonInfo(8, 1);

    return 0;
}
```



Конструкция за избор от варианти switch-case (1)

- При конструкцията switch-case (switch-превключват/преминават към case-случай), в зависимост от конкретната стойност на израз се изпълнява една от множество групи от команди.
- **Синтаксисът на конструкцията** switch-case е следният:

```
switch(<израз>){  
    case <стойност1>: [<команди1> [break;]]  
    case <стойност2>: [<команди2> [break;]]  
    ...  
    case <стойностN>: [<командиN> [break;]]  
    [default: [<команди по подразбиране>]]  
}
```

Конструкция за избор от варианти switch-case (2)

- **Типът на израза** може да бъде от целочислен тип или изброим тип (enum).
- След всяка ключова дума case се задава константа от типа на израза, а след символа ':' се дават съответни команди.
- Константите трябва да са уникални (да не се повтарят).
- **Списъкът от стойности** съдържа поне един елемент.
- **Ако при изпълнение на програмата, стойността на израза съвпадне с някоя от описаните case-стойности, започват да се изпълняват съответните на case-случая команди.**
- **Частта по подразбиране** се означава с default и **не е задължителна**. Тя се изпълнява само ако стойността на израза не съвпадне с никоя от изброените case-стойности.

Конструкция за избор от варианти switch-case (3)

- Командата за прекъсване `break` не е задължителна и се използва, за да се прекъсне изпълнението на следващите групи от команди.
- Тоест: **ако не се зададе `break`, освен групата команди, отговаряща на намерения case-случай, ще се изпълнят и всички останали групи команди** след него.
- Изводът е, че трябва да се внимава каква точно логика трябва да се реализира и да не се забравя за съществуването на `break`.

Конструкция за избор от варианти switch-case (4)

- От синтаксиса на конструкцията се вижда, че **след „case стойност :“ не е задължително да се задават команди**. По този начин може да се опише една и съща група от команди да се отнася за няколко case-стойности.
- **Конструкцията switch-case може да се реализира и чрез множество вложени if-else конструкции**. Когато броят на случаите е много голям, if-else конструкциите може да станат трудни както за описание, така и за четене.

Пример – Информация за месец по зададен номер (1)

```
void monthInfo(int monthNum) {  
    string monthName; // име на месец  
    string quarter; // тримесечие  
    switch (monthNum) {  
        case 1: { monthName = "януари"; quarter = "първо"; break; }  
        case 2: { monthName = "февруари"; quarter = "първо"; break; }  
        case 3: { monthName = "март"; quarter = "първо"; break; }  
        case 4: { monthName = "април"; quarter = "второ"; break; }  
        case 5: { monthName = "май"; quarter = "второ"; break; }  
        case 6: { monthName = "юни"; quarter = "второ"; break; }  
        case 7: { monthName = "юли"; quarter = "трето"; break; }  
        case 8: { monthName = "август"; quarter = "трето"; break; }  
        case 9: { monthName = "септември"; quarter = "трето"; break; }  
        case 10: { monthName = "октомври"; quarter = "четвърто"; break; }  
        case 11: { monthName = "ноември"; quarter = "четвърто"; break; }  
        case 12: { monthName = "декември"; quarter = "четвърто"; break; }  
        default: { monthName = "несъществуващ"; quarter = "неизвестно"; }  
    }  
  
    cout << "Месец с номер " << monthNum << " е " << monthName << " и е в " << quarter << " тримесечие." << endl;  
}
```

Пример – Информация за месец по зададен номер (1)

```
int main()
{
    setlocale(LC_ALL, "bg");

    monthInfo(1);
    monthInfo(7);
    monthInfo(13);

    return 0;
}
```

